Degree competences to which the subject contributes

Specific:
1. CEFB5. Knowledge of informatic systems, its structure, function and interconnection, as well as fundamentals of its programming.
2. CEFC2. Ability to plan, conceive, develop, manage informatic projects, services and systems in all areas, leading their implementation and continuous improvement assessing their economic and social repercussions.
3. CEFC8. Ability to analyze, to design, to construct and to maintain applications in a well built, secure and efficient way choosing the most adequate paradigms and languages.
4. CEFC9. Ability to know, understand and assess computer structure and architecture, as well as basic components forming them.
5. CEFC14. Knowledge and application of fundamentals principals and basic techniques of parallel, concurrent, distributed and real time programming.

Prior skills

It is advisable to have studied Computer Architecture and Operating Systems.

Teaching methodology

Theory classes are conducted using the resources available in the classroom (whiteboards, multimedia equipment) and are based on oral exposure by teachers of content on the subject under study (expository method). In some cases, there will be lectures based on the participation and involvement of students through short-term activities in the classroom, such as direct questioning, student presentations on specific topics or resolution of problems related to the theoretical exposed. Also the teacher will solve classroom exercises and propose collection exercises for students to prepare them independently. These exercises will be solved in class by the students individually or in groups of two people.

Small group classes are:
- Laboratory classes: be performed on school computer classrooms. The student must take practice prepared (read and understand the statement of the practice from a script that was previously found in digital campus), and sometimes if indicated shall make a preliminary report. The practices will be individual.

Learning objectives of the subject

The main objectives of this course are:
- Paradigms concurrency, parallel and distributed systems: (client-server, load sharing, tasks, etc.)..
340384 - PACO-I5001 - Parallelism and Concurrence

- Platforms parallel (shared memory architectures, distributed memory).
- Tools to aid the development of parallel programs.
- Programming and evaluation of parallel programs (programming models for different parallel platforms).
- Memory Coherence and consistency. Communication synchronization, race conditions, mutex, critical section, monitors, deadlock.

**Study load**

<table>
<thead>
<tr>
<th>Total learning time: 150h</th>
<th>Hours large group:</th>
<th>45h</th>
<th>30.00%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hours medium group:</td>
<td>0h</td>
<td>0.00%</td>
</tr>
<tr>
<td></td>
<td>Hours small group:</td>
<td>15h</td>
<td>10.00%</td>
</tr>
<tr>
<td></td>
<td>Guided activities:</td>
<td>0h</td>
<td>0.00%</td>
</tr>
<tr>
<td></td>
<td>Self study:</td>
<td>90h</td>
<td>60.00%</td>
</tr>
</tbody>
</table>
# 1. Introduction to parallelism

**Description:**
Need of parallelism. Parallelism versus concurrence. Problems using concurrence: deadlock, lifelock, starvation, fairness, data races

**Related activities:**
- Activity 1. Unit 1 problems
- Activity 2. Lab 0: Experimental setup, tools and programming model

<table>
<thead>
<tr>
<th>Learning time: 9h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory classes: 1h</td>
</tr>
<tr>
<td>Practical classes: 2h</td>
</tr>
<tr>
<td>Self study: 6h</td>
</tr>
</tbody>
</table>

# 2. Analysis of parallel applications

**Description:**
Can a computation be divided into different parts? It is divided based on the tasks to do or based on the input/output data. Will there be dependence of data between the tasks? How will they be solved? A good decomposition determines the parallel degree achievable.

**Related activities:**
- Activity 1. Problems: Analysis of parallel applications
- Activity 2: Lab 0: Experimental setup, tools and programming model

<table>
<thead>
<tr>
<th>Learning time: 11h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory classes: 1h</td>
</tr>
<tr>
<td>Practical classes: 2h</td>
</tr>
<tr>
<td>Laboratory classes: 2h</td>
</tr>
<tr>
<td>Self study: 6h</td>
</tr>
</tbody>
</table>
### 3. Introduction to (shared-memory) Parallel Architectures

**Learning time:** 20h  
- Theory classes: 2h  
- Practical classes: 4h  
- Laboratory classes: 2h  
- Self study: 12h

**Description:**  
Parallelism inside a processor (IDLP, DLP, TLP), multiprocessors with share memory, multiprocessors with distributed memory.

**Related activities:**  
- Activity 1. Unit 3 problems  
- Activity 2. Lab 0: Experimental setup, tools and programming model  
- Activity 4. Knowledge test

### 4. Basics of parallel programing: Tasks decomposition

**Learning time:** 22h  
- Theory classes: 2h  
- Practical classes: 4h  
- Laboratory classes: 2h  
- Guided activities: 2h  
- Self study: 12h

**Description:**  
Identification of concurrence patterns. Tasks decomposition, granularity and analysis of dependences. Identification of parallelism patterns: task parallelism versus divide and conquer. Mechanisms to implement the task decomposition: thread creation and destruction, thread synchronization patterns, exclusion when accessing share data.

**Related activities:**  
- Activity 1. Task parallelism problems  
- Activity 2. Lab 1: Embarrassingly parallelism with OpenMP: Mandelbrot set
5. Programming with Shared Memory

<table>
<thead>
<tr>
<th>Learning time: 31h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory classes: 3h</td>
</tr>
<tr>
<td>Practical classes: 6h</td>
</tr>
<tr>
<td>Laboratory classes: 2h</td>
</tr>
<tr>
<td>Guided activities: 2h</td>
</tr>
<tr>
<td>Self study: 18h</td>
</tr>
</tbody>
</table>

Description:
Parallel regions, threads and tasks. Task threads, barriers, mutual exclusion locks. Work distributors: loops, sections.

Related activities:
- Activity 1. Share memory problems
- Activity 2. Lab 2: Divide and Conquer parallelism with OpenMP: Sorting
- Activity 3. Directed work. Additional practice

6. Basics of Parallel Programming: Data Decomposition

<table>
<thead>
<tr>
<th>Learning time: 31h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory classes: 3h</td>
</tr>
<tr>
<td>Practical classes: 6h</td>
</tr>
<tr>
<td>Laboratory classes: 4h</td>
</tr>
<tr>
<td>Self study: 18h</td>
</tr>
</tbody>
</table>

Description:
Data decomposition (geometric versus recursive structure), data flow organization (regular versus irregular). Mechanisms to implement the data decomposition: creation and destruction process, process synchronization (barrier) and communications patterns (point-to-point communication, synchronous and asynchronous communication).

Related activities:
- Activity 1. Data decomposition problems
- Activity 2. Lab 3: Geometric decomposition: solving the heat equation

Qualification System

1st partial knowledge test *0.2 + problems *0.1 + 0.3* Laboratory + complementary *0.1 Working + 2nd partial knowledge test *0.3* = 5
The 1st and 2nd partial knowledge tests are reevaluable

Regulations for carrying out activities

Activities 1, 2 and 4 are in person.
Activity 3 is non-attendance, although there may be a short presentation in class.
In the activities that take place in group the mark will be the same for all group members.
340384 - PACO-I5001 - Parallelism and Concurrence

Bibliography

**Basic:**


**Others resources:**

- **Computer material**
  - **Software a Boada**
    - Connection and software in boada.ac.upc.edu